

# OCR A Level

## Computer Science

### H446 – Paper 2



## Bubble sort and insertion sort

Unit 12  
Algorithms



**PG ONLINE**

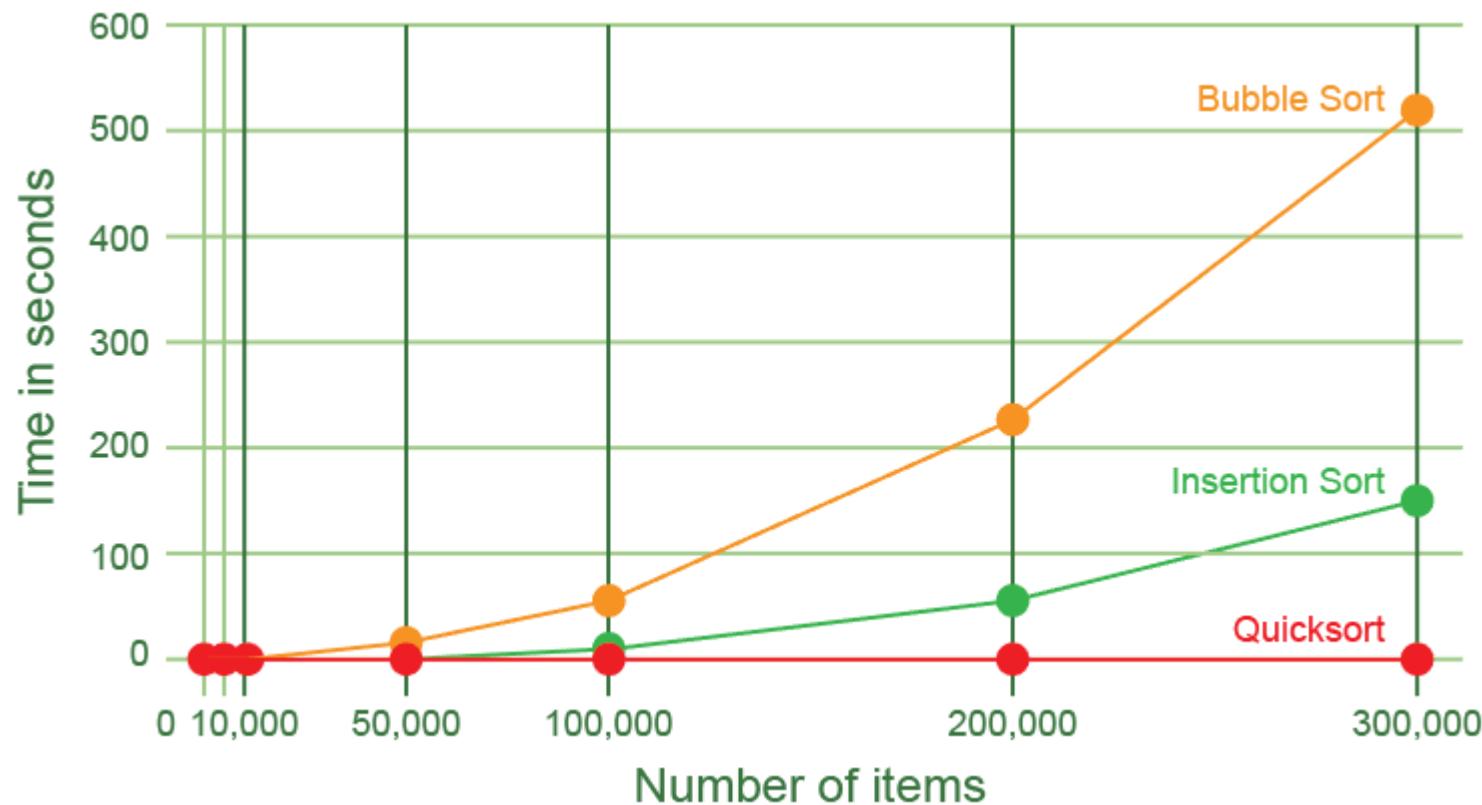
# Objectives

- Be able to describe the bubble sort and insertion sort algorithms
- Be able to trace the bubble sort and insertion sort algorithms

# Sorting algorithms

- Sorting is a very common task in data processing
- Frequently the number of items to be sorted is huge. So it is important to choose an efficient sorting algorithm
- The **quicksort** is nearly 3000 times faster than the **bubble** sort for sorting 100,000 items!

# Time performance of sorting algorithms



# Choosing a sort algorithm

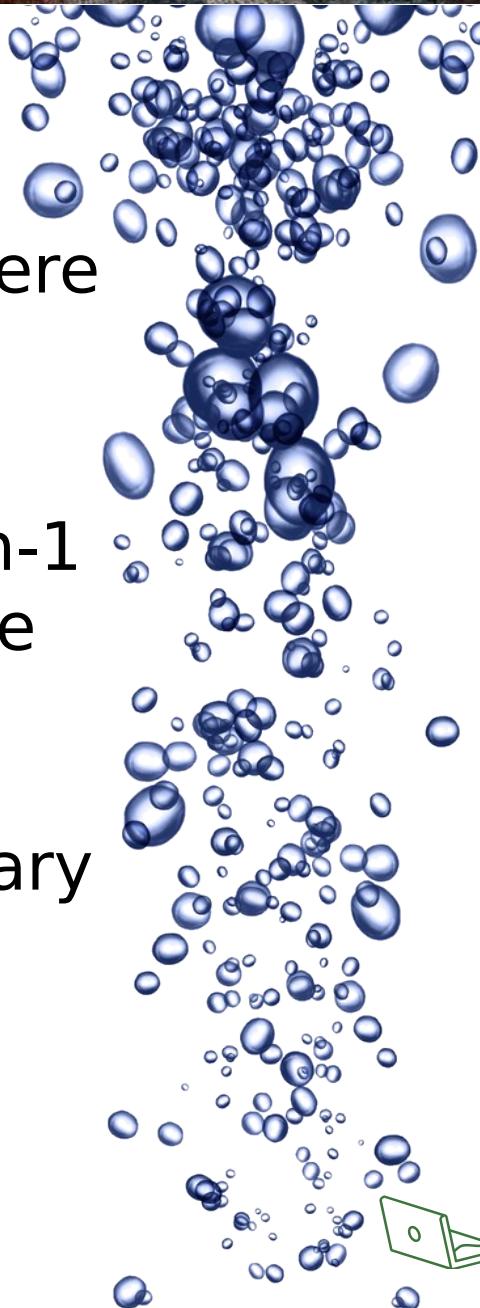
- Why would anyone choose a **bubble sort** when a **quicksort** is so much faster?
- There are international competitions to design the fastest sort algorithm...

# Choosing a sort algorithm

- If there are only a small number of items to sort, a bubble sort may be perfectly adequate
- 0.6 of a second to sort 10,000 items sounds reasonable
- A week to sort 3 million items – not so good!
- Also, some sort algorithms take a lot of memory and that may be a problem with a large dataset

# Bubble sort

- This is a useful sort when there are only a small number of items to be sorted
- To sort an array of  $n$  items,  $n-1$  passes are made through the array, with each item being compared with the adjacent item and swapped if necessary



# Bubble sort algorithm

```
for i = 0 to n - 2
    for j = 0 to (n - i - 2)
        if a [j] > a[j + 1]
            swap the items
    endif
    next j
next i
```

- Trace through the algorithm for  $a[7,4,6,8,1,5]$
- Write the code for the line “swap the items”

# Analysing the bubble sort

- How many statements are there in the worst case scenario, in which a swap is made every time?
- Assume that “swap the items” counts as three statements

```
for i = 0 to n - 2
    for j = 0 to (n - i - 2)
        if names[j] > names[j + 1]
            swap the names
        endif
    next j
next i
```

# Big-O for bubble sort

- The three statements in the inner nested loop are performed  $n(n-1 + n-2 + n-3 + \dots + 1)$
- This is approximately  $3 \times \frac{1}{2} n^2$  statements
- Ignoring the constant, time complexity =  $O(n^2)$

```
for i = 0 to n - 2
    for j = 0 to (n - i - 2)
        if names[j] > names[j + 1]
            swap the names
        endif
    next j
next i
```

# Worksheet 3

- Do **Task 1** on Worksheet 3



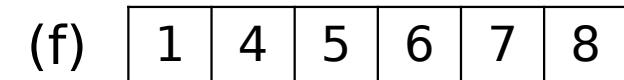
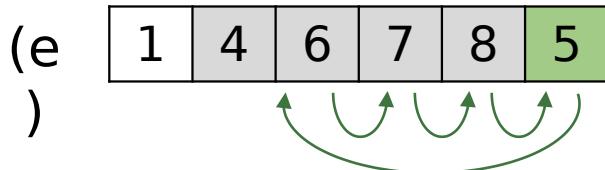
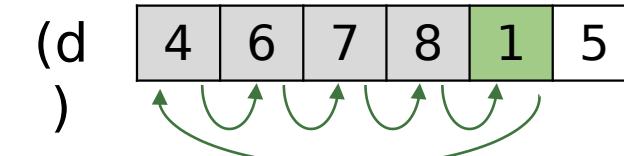
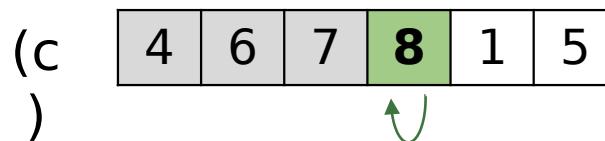
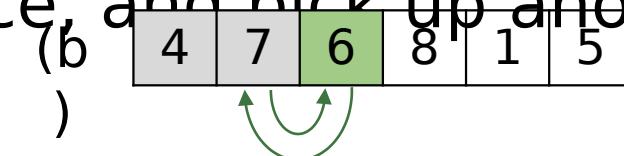
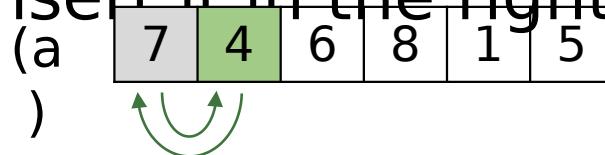
# Insertion sort

- The insertion sort is an efficient algorithm for sorting a small number of elements
- It works in the same way you might sort a deck of playing cards
  - Start with all the cards face down
  - Pick up each card in turn and insert it into the correct place in the deck



# How the Insertion sort works

- Suppose we have six numbers to sort
- The green card is the card you pick up each time
- Insert it in the right place, and pick up another



# Algorithm for insertion sort

```
procedure insertionSort(aList)
    for j = 1 to len(aList) - 1
        nextItem = aList[j]
        i = j - 1
        while i >= 0 and aList[i] >
nextItem
            aList[i + 1] = aList[i]
            i = i - 1
        endwhile
        aList[i + 1] = nextItem
        next j
endprocedure
```

# Worksheet 3

- Do **Task 2** on Worksheet 3



# Time complexity of insertion sort

- Look again at the algorithm for the insertion sort
  - How many loops does it have?
  - Does it have nested loops?
  - How many times is each loop performed?
  - Can you work out the Big O time complexity?

# Time complexity of insertion sort

- The insertion sort algorithm has two nested loops
- The outer loop is performed  $n-1$  times
- The inner loop, which moves the items along to make room for the current item, is performed a maximum of  $n-1$  times
- The Big-O time complexity  $O(n^2)$

# Plenary

- You need to be able to describe how the bubble sort and the insertion sort are carried out
- They both have the same time complexity  $O(n^2)$  but the insertion sort is generally faster, although of the same order - e.g. the bubble sort may take twice as long to sort 10,000 items

## Copyright

© 2016 PG Online Limited

The contents of this unit are protected by copyright.

This unit and all the worksheets, PowerPoint presentations, teaching guides and other associated files distributed with it are supplied to you by PG Online Limited under licence and may be used and copied by you only in accordance with the terms of the licence. Except as expressly permitted by the licence, no part of the materials distributed with this unit may be used, reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic or otherwise, without the prior written permission of PG Online Limited.

## Licence agreement

This is a legal agreement between you, the end user, and PG Online Limited. This unit and all the worksheets, PowerPoint presentations, teaching guides and other associated files distributed with it is licensed, not sold, to you by PG Online Limited for use under the terms of the licence.

The materials distributed with this unit may be freely copied and used by members of a single institution on a single site only. You are not permitted to share in any way any of the materials or part of the materials with any third party, including users on another site or individuals who are members of a separate institution. You acknowledge that the materials must remain with you, the licencing institution, and no part of the materials may be transferred to another institution. You also agree not to procure, authorise, encourage, facilitate or enable any third party to reproduce these materials in whole or in part without the prior permission of PG Online Limited.